Dalek: An Unconventional and Energy-Aware Heterogeneous Cluster

Adrien Cassagne, Noé Amiot and Manuel Bouyer

September 8th, 2025

Table of Contents

A Cluster Designed for Researchers

- A cluster composed by **heterogeneous nodes**
- Made from low power mini-PC nodes: What an idea?!
  - Absolutely not designed for clustering ➔ **Challenging**!
- Advantages
  - Components from the public at large ➔ **Cheap**!
  - **Fast availability** of the chips and **easy to upgrade**
- For who?
  - LIP6 researchers
  - **CS architecture enthusiasts** in general

Description of the Partitions

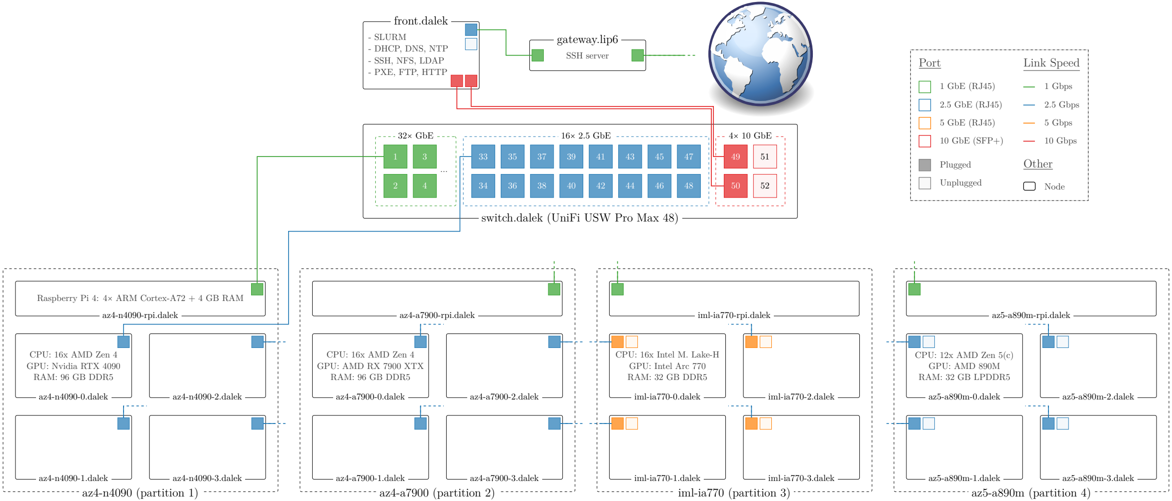
Partition Name	CPU	RAM	GPU	Amount of Resources: Nodes	Cores	HW Threads	RAM (GB)	Power Consumption: Idle (W)	Suspend (W)	TDP (W)
az4-n4090 (partition 1)	AMD Ryzen 9 7945HX (16x Zen 4 cores)	96 GB DDR5	Nvidia GeForce RTX 4090	4	64	128	384	212	6	2100
az4-a7900 (partition 2)	AMD Ryzen 9 7945HX (16x Zen 4 cores)	96 GB DDR5	AMD Radeon RX 7900 XTX	4	64	128	384	192	6	1500
iml-ia770 (partition 3)	Intel Core Ultra 9 185H (16x Meteor Lake-H cores)	32 GB DDR5	Intel Arc A770	4	64	88	128	260	92	1360
az5-a890m (partition 4)	AMD Ryzen AI 9 HX 370 (12x Zen 5 cores)	32 GB LPDDR5x	AMD Radeon 890M	4	48	96	128	16	8	216
front	Intel Core i9-13900H (14x Raptor Lake-H cores)	96 GB DDR5	Intel Iris Xe Graphics	1	14	20	96	15	–	115
switch	Unifi Pro Max 48	–	–	–	–	–	–	20	–	100
Total	–	–	–	**17**	**254**	**460**	**1120**	**715**	**112**	**5391**

Infrastructure

- **Half rack 25U** on wheels with glass for demonstrations
- Network: Unifi Switch Pro Max **48 ports**
  - 4 × SFP+ 10 Gbps (optical fiber)
  - 16 × Ethernet 2.5 Gbps
  - 32 × Ethernet 1 Gbps
- **Frontend node**
  - 14 cores Intel Core i9-13900H, 96 GB DDR5
  - 2 × Ethernet 2.5 Gbps ports
  - 2 × SFP+ 10 Gbps ports
  - 6 TB PCIe 4 NVMe SSD for NFS
- Energy consumption (estimated)
  - Idle: 715 Watts
  - Maximum: **5391 Watts** (requires two standard sockets)

Cluster Topology – The Big Picture

1 Dalek Presentation



How does it Look in Real?

From the front it is pretty clean!

How does it Look in Real?

az4-n4090 (partition 1)

- Vertical & **open ITX PC case**
- 1000 W SFX power supply
- **Nvidia RTX 4090** through PCIe 4

From the front it is pretty clean!

How does it Look in Real?

az4-a7900 (partition 2)

- Vertical & open ITX PC case
- **1000 W SFX power supply**
- AMD 7900 XTX through **PCIe 4**

From the front it is pretty clean!

How does it Look in Real?

iml-ia770 (partition 3)

- Intel-based **Mini PC** (low power)
- External GPU through **Oculink**!
  - 1000 W SFX power supply
  - Intel Arc 770 GPU

From the front it is pretty clean!

How does it Look in Real?

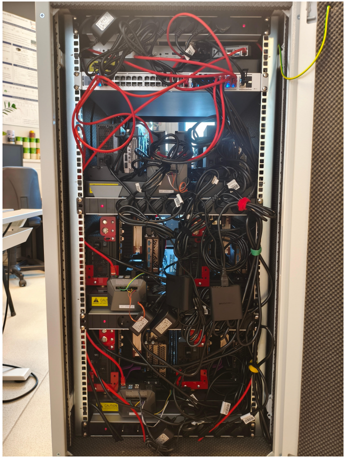
az5-a890m (partition 4)

- AMD-based **Mini PC** (low power)
- Last **Zen 5** & Zen 5c CPU arch.
- Integrated Radeon 890M GPU

From the front it is pretty clean!

How does it Look in Real?

From the front it is pretty clean!



From the back... :-)

Table of Contents

The Challenges

The biggest challenge is to be able to **administrate a cluster composed of computers that are not dedicated to clustering**.

1. Which **distribution** to choose to have **recent Linux kernels** and drivers?
2. How to automatically and **remotely install nodes** when needed?
3. How to manage **resources allocation** in a multi-user environment?
4. How to keep **local user data** even after a re-installation?
5. How to manage **power up/down** of compute nodes without IPMI?

Ubuntu is for Beginners... Really?!

- **Ubuntu** releases recent kernel twice a year
- Light **server versions** are available
- Well documented and **tested over millions of laptop**
  — Dalek's compute nodes come from laptop
- When possible **LTS versions are preferred**
- Many packages are available and easy to install
  — Combined with **Module Environment** when needed

```
$ module avail
-------- /mnt/nfs/software/modules-env/etc/modulefiles/software --------
aff3ct/3.0.2-152-g60b147a  mipp/g75fc843  streampu/1.6.1-44-g13feed3

-------- /mnt/nfs/software/modules-env/etc/modulefiles/compiler --------
pocl/7.0
```

Ubuntu Autoinstall and PXE

- Ubuntu comes with a dedicated tool to **manage auto installations**
  - Administrators define packages, disk partitioning and are free to execute early and late commands to customize the installation even more
- **PXE** (≈ boot from the network) make possible to decide whether
  - To boot on local drive (normal behavior)
  - To install a new version of Ubuntu
- Ubuntu image is served via a light FTP server
  - Ubuntu versions and packages depend on the partition
  - ➔ Particularly useful for drivers installation
- Administrators can connect to nodes during installation via SSH
  - Logs are pushed on the front node via `systemd-journal-remote`
  - ➔ It takes **around 15 minutes to (re-)install the 16 nodes** at the same time

SLURM for Resource Management

- **Standard** to manage resources allocation in HPC clusters
- **Highly configurable** and flexible
  - **SPANK** make it easy to add custom plugins
- Standard commands and batch scripts are supported (`sinfo`, `squeue`, `srun`, `salloc`, `sbatch`, `scontrol`, ...)

```
$ sinfo
PARTITION  AVAIL  TIMELIMIT  NODES  STATE NODELIST
az4-n4090     up   infinite      4  idle~ az4-n4090-[0-3]
az4-a7900     up   infinite      4  idle~ az4-a7900-[0-3]
az4-mixed     up   infinite      8  idle~ az4-a7900-[0-3],az4-n4090-[0-3]
iml-ia770     up   infinite      4  idle~ iml-ia770-[0-3]
az5-a890m     up   infinite      4  idle~ az5-a890m-[0-3]
```

Scratch

- In HPC a scratch is a mount point on the local drive
  - **Faster than the network file system** (NFS) but local to one node
- Scratch **folder is automatically created** at the first login of an user
  - Achieved by a SLURM SPANK plugin that invokes a PAM script
- **Scratch folder is persistent**
  - Data remains after logout
  - Data remains after a re-installation
  - ➔ **Not common at all** in HPC environments
- The scratch is a specific partition created or kept (if already exists) during the Ubuntu Autoinstall process

Powering Up and Down Compute Nodes

- **Power up**
  - Compute nodes are configured to wait for the **magic packet** on their network interface (Wake on Lan standard)
  - SLURM `noderesume` script has been configured
- **Power down**
  - A **special user** (`powerstate`) is automatically created on each compute node
  - `powerstate` can execute `poweroff` command without entering its password
  - SLURM `nodesuspend` script has been configured
- SLURM has been configured to automatically run the `nodesuspend` script on a compute node after **10 minutes of inactivity**
- When a user requires nodes, if they are powered off, SLURM automatically wake up the node through the `noderesume` script
  - **Waking up** can take up to **1 minute and 30 seconds**

Table of Contents

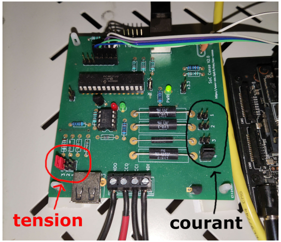
3 Power Measurement Platform

Proposal

- Custom platform to **measure energy consumption in real time**
- Capable of 2000+ samples per second
  - Users will get **1000 averaged samples per second** at the milliWatt resolution
  - In comparison, Grid'5000 cluster is able to give 50 samples per second at the resolution of half a Watt and from the socket (220 V)[1]
- Proposed platform will measure energy consumption from the **continuous current** (19 V, 12 V, 5 V and 3.3 V)
  - Make possible to isolate components (mother board, CPU, GPU and drives)
- Same platform will be able to measure energy consumption from **USB-C Power Delivery** standard, up to 240 Watts (for mini-PCs and SBCs)

[1] Grid'5000: https://www.grid5000.fr/w/Energy_consumption_monitoring_tutorial

Previous Platform for the Monolithe

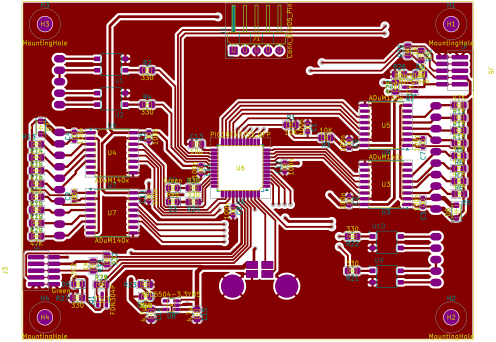


- A first version of the measurement platform is already deployed in the Monolithe cluster[a]
- **Measure power from the socket** up to 95 Watts
- **GPIO** to be able to send signal from the running app.
- **Control power state** from the platform **remotely** (≈ IPMI)

[a] A. Cassagne, L. Lacassagne, M. Bouyer, et al. *Monolithe*. Web page: https://monolithe.proj.lip6.fr. 2022.

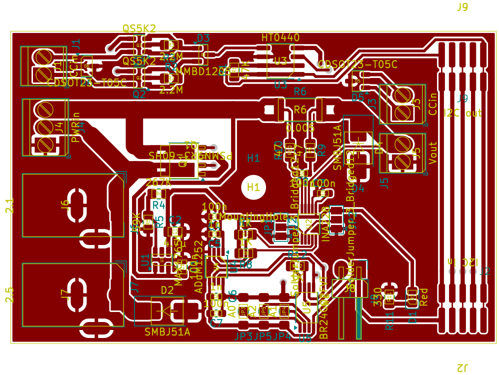
New Modular Platform – Main Board

- One per compute node and based on **PIC 18 micro-controller**
- Connect to a computer to **output samples via USB**
- Accept up to **two probe chains**
  - — 6 probes can be **chained together and plugged** to a main board connector
- Powered via USB (5 V)

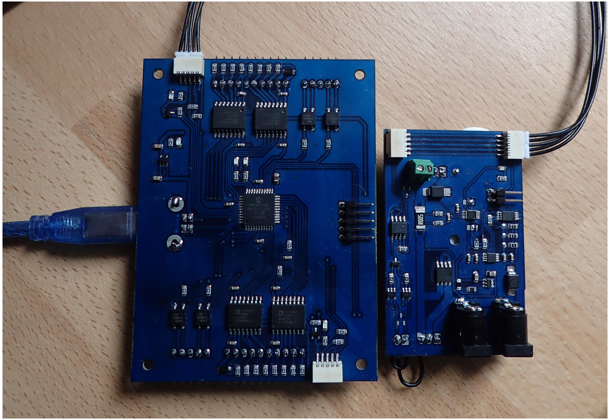


New Modular Platform – Probes

- In-between power input and the node
- Transfer **measured samples through I2C bus** to the main board
- **Modular design** to support multi-mode power collection
  - USB-C PD (up to 240 W)
  - Coaxial connectors: 5 V, 12 V, 19 V
  - PC PSU Molex connectors: 3.3-5-12 V
  - 600 W PCIe 5.0 12VHPWR (for GPU)
  - Can include **heat** and **humidity** sensors



New Modular Platform – Working Prototype



To the best of our knowledge, this is the first **Open Hardware** and **Open Source API** solution that combines **modular design** and **high resolution sampling**!

Table of Contents

4 Synthetic Benchmarks

CPU Memory Throughput

- Memory throughput is **the limiting factor in most applications**
  - The number of **bytes per second** (10^9 B/s = 1 GB/s)
- Throughput is measured with `bandwidth`[a]
  - Simple (**but useful**) tool developed in the LIP6 ALSOC team
  - Inspired by the famous HPC `STREAM` benchmark
  - Open source and available on GitHub

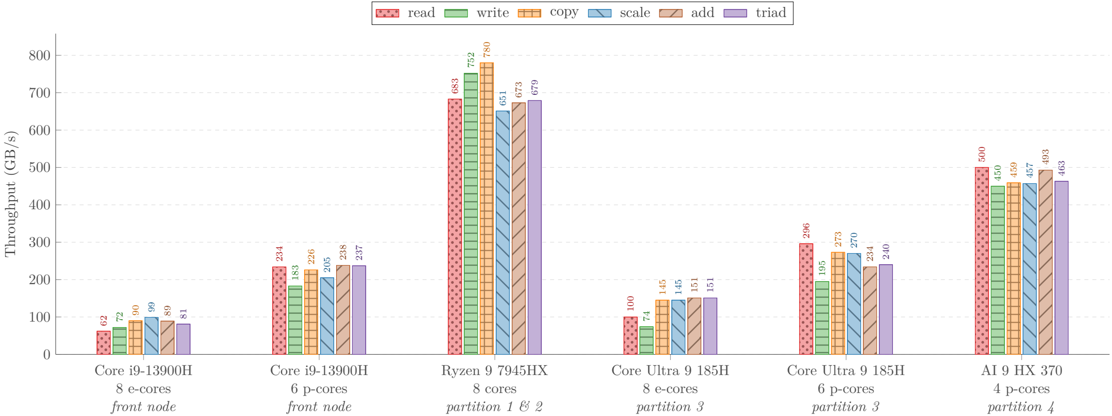
- Simple kernels over varying buffer size

```
  read x = A[i]
 write A[i] = x
  copy B[i] = A[i]
 scale B[i] = x * A[i]
   add C[i] = A[i] + B[i]
triadd C[i] = x * A[i] + B[i]
```

[a] F. Lemaitre and L. Lacassagne. *bandwidth*. GitHub: `https://github.com/alsoc/bandwidth`. 2017.

CPU Memory Throughput – L3 Cache

Measurements with the **`bandwidth`** benchmark. Buffer of 20 MB. **Higher is better**.



CPU Peak Performance

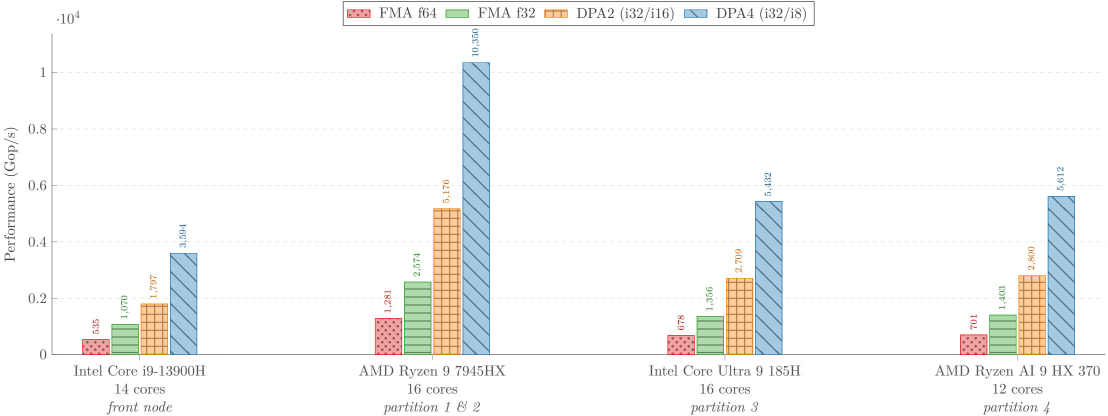
- CPU peak performance is the number of operations that the CPU can do in a given amount of time
  - Generally accounted as the number of **operation per second**
  - 10^9 op/s = 1 Gop/s
- Peak performance is measured with the `cpufp`[a] benchmark
  - Stress the CPU at its limit
  - **Assembly code** that does not aim to perform realistic computations
  - Open source and available on GitHub

- Driving performance instructions
  - FMA: Fused Multiply-Add
    - In most of the current architectures
    - Floating-point comp. (32-bit to 64-bit)
    - $d = a \times b + c$
  - DPA2: 2-way Dot Product Accumul.
    - Available through VNNI ext. (for AI)
    - Fixed-point mixed prec. (16-bit/32-bit)
    - $c^{i32} = c^{i32} + \sum_{s=1}^{2} a_s^{i16} \times b_s^{i16}$
  - DPA4: 4-way Dot Product Accumul.
    - Available through VNNI ext. (for AI)
    - Fixed-point mixed prec. (8-bit/32-bit)
    - $c^{i32} = c^{i32} + \sum_{s=1}^{4} a_s^{i8} \times b_s^{i8}$

[a] Y. Gao et al. *cpufp*. GitHub: `https://github.com/pigirons/cpufp`. 2016.

CPU Peak Performance – Multi-core Accumul.

Measurements with the `cpufp` benchmark. **Higher is better**.



GPU Peak Performance

- **GPU peak is generally crazy high compared to CPUs**
  - Using Top/s (= 10^{12} op/s) instead of Gop/s
- Peak performance is measured with `clpeak`[a]
  - Target only the general compute cores
  - Tensor and ray tracing cores are not evaluated
  - Open source and available on GitHub

- GPUs performance varies a lot depending on the datatypes
  - `float16`
  - `float32`
  - `float64`
  - `int8`
  - `int16`
  - `int32`

[a] K. Bhat et al. *clpeak*. GitHub: `https://github.com/krrishnarraj/clpeak`. 2013.

GPU Peak Performance – Shader Cores

Measurements with the `clpeak` benchmark. **Higher is better**.

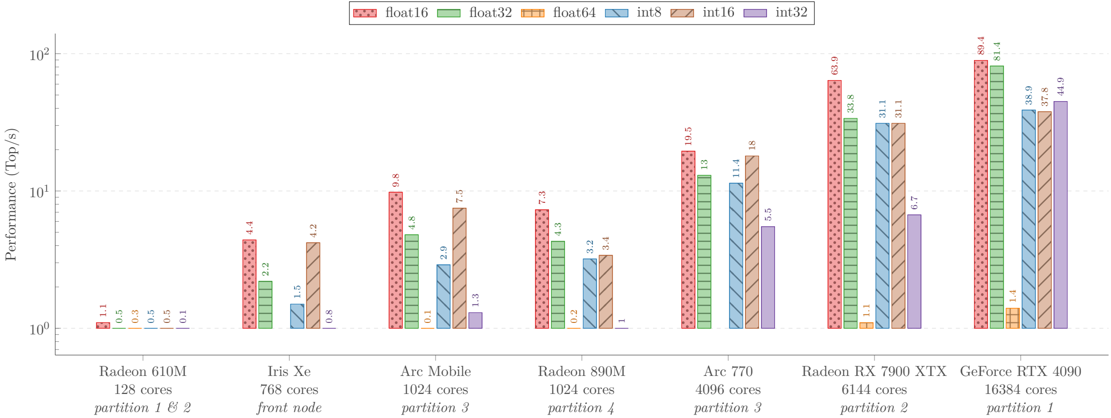


Table of Contents

Conclusion

5 Final Words

- DALEK is **already working** but still in **testing phase**
  — Ask me directly for an account creation
- Remaining things
  — **Install MPI** implementations (Open MPI, MPICH)
  — Evaluate the network performance
  — Setup SLURM **queues with time limit**
- **Documentation** available online: `https://dalek.proj.lip6.fr`
- **Preprint document** about DALEK available on **HAL** and **arXiv**[1]
  — Please **cite this document** in your research papers if you are using DALEK
  ➔ Enable **traceability** and **visibility** of the platform

[1] Adrien Cassagne, Noé Amiot, and Manuel Bouyer. “Dalek: An Unconventional and Energy-Aware Heterogeneous Cluster”. (preprint). Aug. 2025. DOI: 10.48550/arXiv.2508.10481.

Acknowledgments

This study has been carried out with financial support from the French State, managed by the French "Agence Innovation Défense" and for the DGA-Inria convention around the AFF3CT [6] project.

Special thanks to Pierre Tercinet and Pierre Loidreau who believed in this ambitious project and allowed us to make it real.

Thanks also to the LIP6 direction for your guidance and help. First, thank you Francis Hulin-Hubard for supporting the project since the beginning. And, of course, special thanks to Hélène Petridis, Aline Levaillant and Fabrice Kordon for believing in the project and helping us to achieve it, sincerely.

Q&A

Thank you for listening!
Do you have any questions?

SCAN ME

Partition Details

- 4 **heterogeneous partitions** of **4 nodes** each
- **NPUs** and some **iGPUs** are not detailed below

	Processor (CPU)					System RAM Memory			
	Vendor	Model	Archi.	Cores	TDP	Type	MT/s	Chn.	Amount
Partition 1 Partition 2	AMD	Ryzen 9 7945HX	Zen 4	16 (32)	75 W	DDR5	5200	2	96 GB
Partition 3	Intel	Core Ultra 9 185H	Meteor Lake	16 (22)	115 W	DDR5	5600	2	32 GB
Partition 4	AMD	Ryzen AI 9 HX 370	Zen 5	12 (24)	54 W	LPDDR5X	7500	4	32 GB

	Graphical Process Unit (GPU)							
				VRAM Memory				
	Vendor	Model	Archi.	Type	Bus	Amount	Cores	TDP
Partition 1	Nvidia	GeForce RTX 4090	Ada Lovelace	GDDR6X	384-bit	24 GB	128	450 W
Partition 2	AMD	Radeon RX 7900 RTX	RDNA 3	GDDR6	320-bit	24 GB	84	300 W
Partition 3	Intel	Arc A770	Alchemist	GDDR6	256-bit	16 GB	32	225 W
Partition 4	AMD	Radeon 890M	RDNA 3.5	*Unified with CPU RAM*			16	–

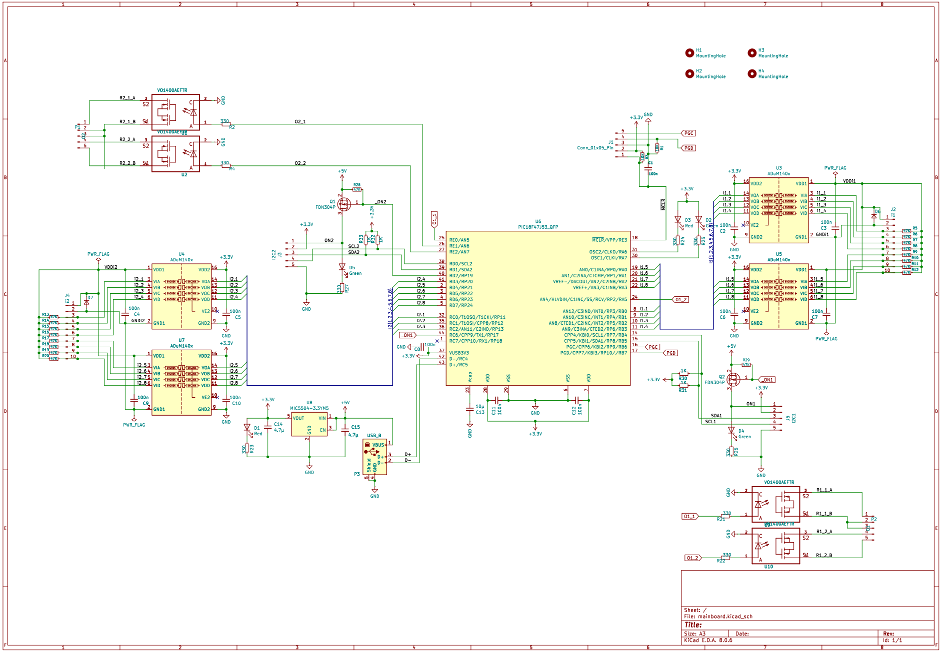
SLURM sinfo States

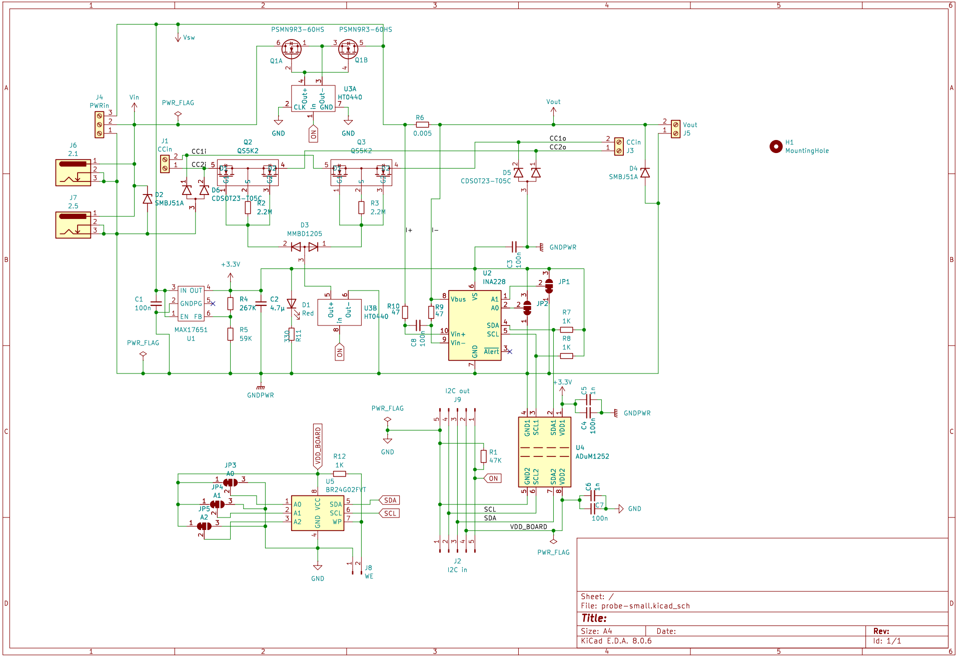
`sinfo` command indicates the node state with the following additional codes:

- `~` The node is presently in powered off
- `#` The node is presently being powered up or configured
- `!` The node is pending power down
- `%` The node is presently being powered down
- `@` The node is pending reboot

```
$ sinfo
PARTITION  AVAIL  TIMELIMIT  NODES  STATE NODELIST
az4-n4090     up   infinite      4  idle~ az4-n4090-[0-3]
az4-a7900     up   infinite      4  idle~ az4-a7900-[0-3]
az4-mixed     up   infinite      8  idle~ az4-a7900-[0-3],az4-n4090-[0-3]
iml-ia770     up   infinite      4  idle~ iml-ia770-[0-3]
az5-a890m     up   infinite      4  idle~ az5-a890m-[0-3]
```

For instance, here all the nodes are powered off.


VO1400AEFTR
U2
R2_1_A
R2_1_B
R2_2_A
R2_2_B
P1
O2_1
O2_2
+5V
R28
47K
Q1
FDN304P
_ON2
ON2
+3.3V
I2C2
J3
SCL2
SDA2
R33
R32
1K
D5
Green
R27
330
PWR_FLAG
U4
ADuM140x
VDDI2
GNDI2
J4
I2
D7
C4
100n
C5
GND
U7
C9
C10
U6
PIC18F47J53_QFP
RE0/AN5
RE1/AN6
RE2/AN7
RD0/SCL2
RD1/SDA2
RD2/RP19
RD3/RP20
RD4/RP21
RD5/RP22
RD6/RP23
RD7/RP24
RC0/T1OSO/T1CKI/RP11
RC1/T1OSI/CPPB/RP12
RC2/AN11/C2IND/RP13
RC6/CPP9/TX1/RP17
RC7/CPP10/RX1/RP18
VUSB3V3
D-/RC4
D+/RC5
MCLR/VPP/RE3
OSC2/CLKO/RA6
OSC1/CLKI/RA7
AN0/C1INA/RP0/RA0
AN1/C2INA/CTCMP/RP1/RA1
VREF-/DACOUT/AN2/C2INB/RA2
VREF+/AN3/C1INB/RA3
AN4/HLVDIN/C1INC/SS/RCV/RP2/RA5
AN12/C3IND/INT0/RP3/RB0
AN10/C3INC/INT1/RP4/RB1
AN8/CTED1/C2INC/INT2/RP5/RB2
AN9/C3INA/CTED2/RP6/RB3
CPP4/KBI0/SCL1/RP7/RB4
CPP5/KBI1/SDA1/RP8/RB5
PGC/CPP6/KBI2/RP9/RB6
PGD/CPP7/KBI3/RP10/RB7
Vcap
VDD
VSS
C11
C12
C13
10µ
_ON1
C8
U8
MIC5504-3.3YM5
VOUT
VIN
EN
C14
C15
4.7µ
D1
Red
R23
USB_B
VBUS
Shield
P3
D+
D-
J1
Conn_01x05_Pin
PGC
PGD
MCLR
R1
C1
D3
D2
R24
R25
H1
H2
H3
H4
MountingHole
U3
U5
VDDI1
GNDI1
C2
C3
C6
C7
J2
I1
D6
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
R16
R17
R18
R19
R20
Q2
R29
R30
R31
ON1
SDA1
SCL1
J5
I2C1
D4
R26
U9
U10
R1_1_A
R1_1_B
R1_2_A
R1_2_B
O1_1
O1_2
R21
R22
P2
Sheet: /
File: mainboard.kicad_sch
Title:
Size: A3
Date:
Rev:
KiCad E.D.A. 8.0.6
Id: 1/1


PSMN9R3-60HS
Q1A
Q1B
U3A
HT0440
Vsw
J4
PWRin
Vin
PWR_FLAG
GND
ON
Vout
R6
0.005
J5
J3
CCin
CC1o
CC2o
J1
CCin
CC1i
CC2i
Q2
QS5K2
Q3
QS5K2
J6
2.1
J7
2.5
D2
SMBJ51A
D6
CDSOT23-T05C
R2
2.2M
R3
2.2M
D3
MMBD1205
D5
CDSOT23-T05C
D4
SMBJ51A
H1
MountingHole
C3
100n
GNDPWR
U2
INA228
JP1
JP2
R7
1K
R8
1K
R9
47
R10
47
C8
100n
+3.3V
C1
100n
U1
MAX17651
R4
267K
R5
59K
C2
4.7µ
D1
Red
R11
330
U3B
HT0440
I+
I-
Vbus
VS
A1
A0
SDA
SCL
Alert
Vin+
Vin-
GND
I2C out
J9
J2
I2C in
R1
47K
U4
ADuM1252
C4
100n
C5
1n
C6
1n
C7
100n
SCL
SDA
VDD_BOARD
JP3
A0
JP4
A1
JP5
A2
U5
BR24G02FVT
R12
1K
J8
WE
Sheet: /
File: probe-small.kicad_sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. 8.0.6
Id: 1/1

CPU Memory Throughput

- Memory throughput is **the limiting factor in most applications**
  — Thus, it is driving the CPU performance
- Throughput is measured with **`bandwidth`**[a]
  — Simple (**but useful**) tool developed in the LIP6 ALSOC team
  — Inspired by the famous HPC `STREAM` benchmark
  — Open source and available on GitHub

- Simple kernels over varying buffer size

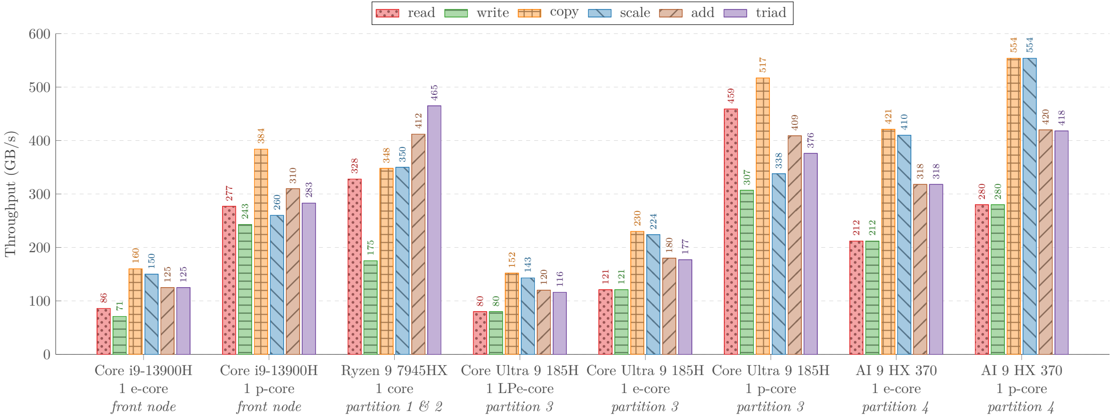
```
  read x = A[i]
 write A[i] = x
  copy B[i] = A[i]
 scale B[i] = x * A[i]
   add C[i] = A[i] + B[i]
triadd C[i] = x * A[i] + B[i]
```

[a] F. Lemaitre and L. Lacassagne. *bandwidth*. GitHub: `https://github.com/alsoc/bandwidth`. 2017.

CPU Memory Throughput – L1D Cache

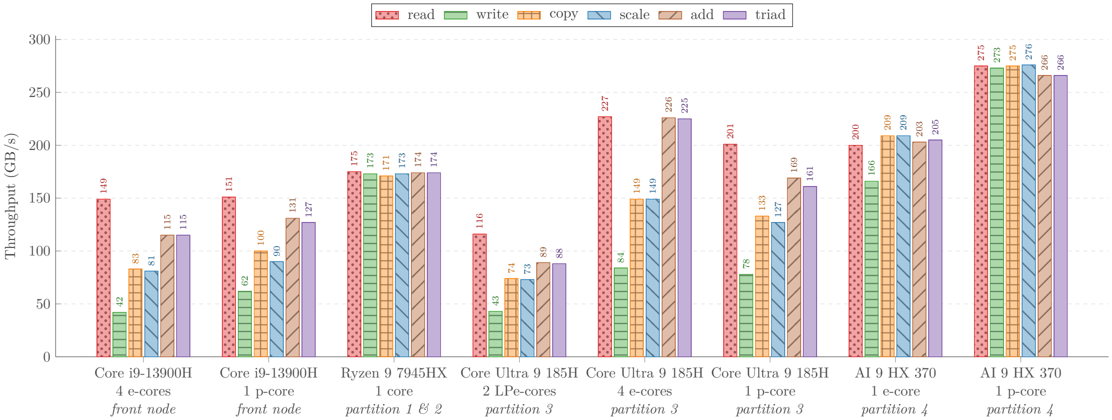
9 Synthetic Benchmarks – Backup Slides

Measurements with the `bandwidth` benchmark. Buffer of 16 KB. **Higher is better**.



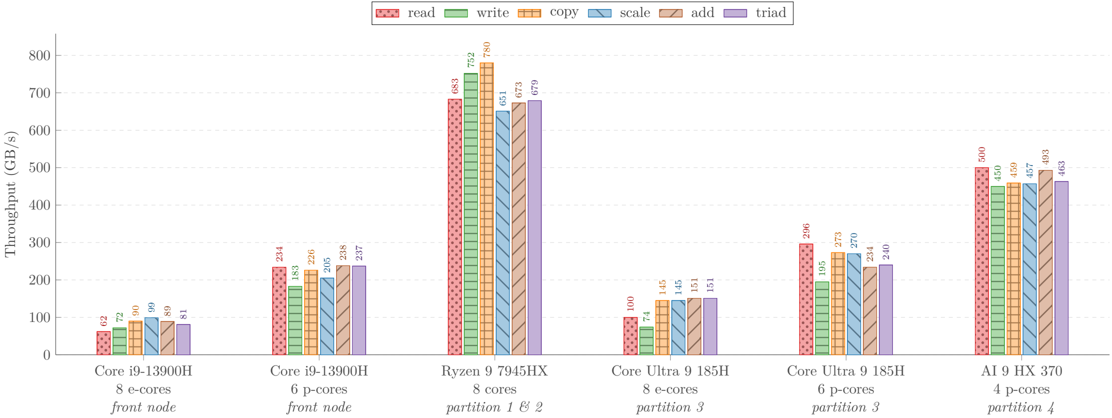
CPU Memory Throughput – L2 Cache

Measurements with the `bandwidth` benchmark. Buffer of 512 KB. **Higher is better**.



CPU Memory Throughput – L3 Cache

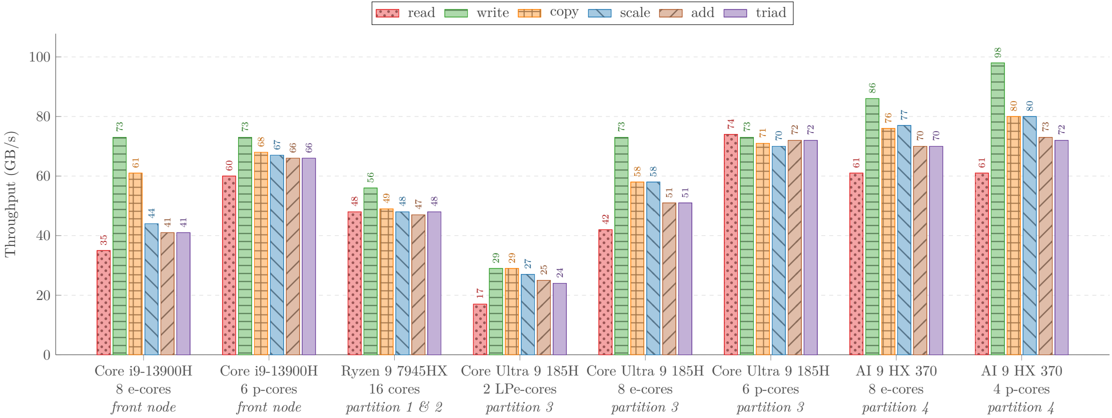
Measurements with the **bandwidth** benchmark. Buffer of 20 MB. **Higher is better**.



CPU Memory Throughput – RAM

9 Synthetic Benchmarks – Backup Slides

Measurements with the **bandwidth** benchmark. Buffer of 2 GB. **Higher is better**.



CPU Peak Performance

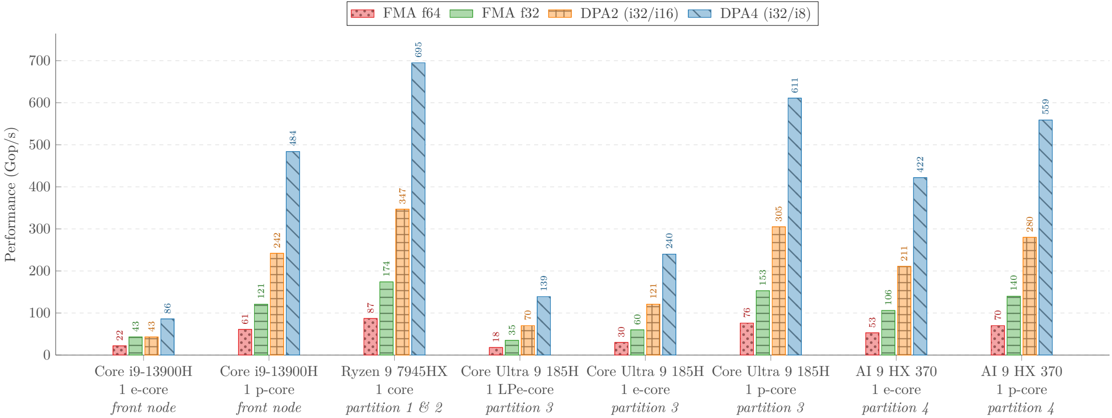
- CPU peak performance is the number of operations that the CPU can do in a given amount of time
  - Generally accounted as the number of **operation per second**
  - 10^9 op/s = 1 Gop/s
- Peak performance is measured with the `cpufp`[a] benchmark
  - Stress the CPU at its limit
  - **Assembly code** that does not aim to perform realistic computations
  - Open source and available on GitHub

- Driving performance instructions
  - FMA: Fused Multiply-Add
    - In most of the current architectures
    - Floating-point comp. (32-bit to 64-bit)
    - $d = a \times b + c$
  - DPA2: 2-way Dot Product Accumul.
    - Available through VNNI ext. (for AI)
    - Fixed-point mixed prec. (16-bit/32-bit)
    - $c^{i32} = c^{i32} + \sum_{s=1}^{2} a_s^{i16} \times b_s^{i16}$
  - DPA4: 4-way Dot Product Accumul.
    - Available through VNNI ext. (for AI)
    - Fixed-point mixed prec. (8-bit/32-bit)
    - $c^{i32} = c^{i32} + \sum_{s=1}^{4} a_s^{i8} \times b_s^{i8}$

[a] Y. Gao et al. *cpufp*. GitHub: `https://github.com/pigirons/cpufp`. 2016.

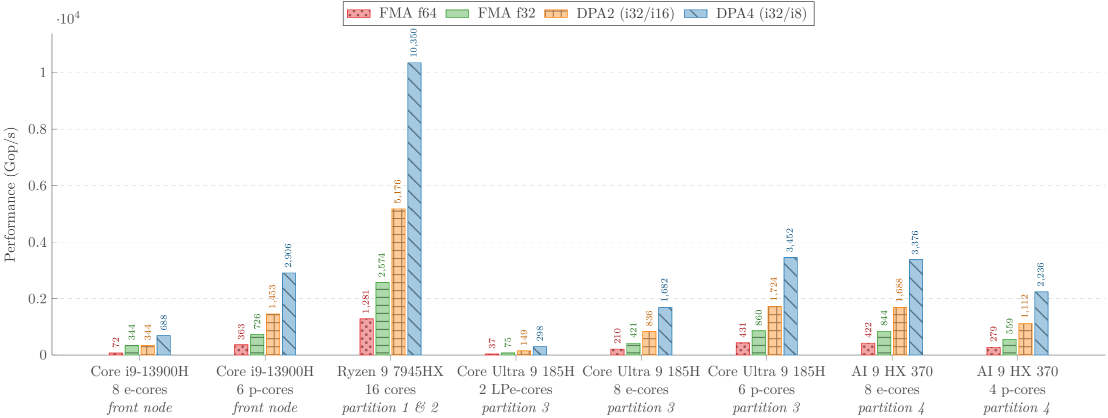
CPU Peak Performance – Single-core

Measurements with the **cpufp** benchmark. **Higher is better**.



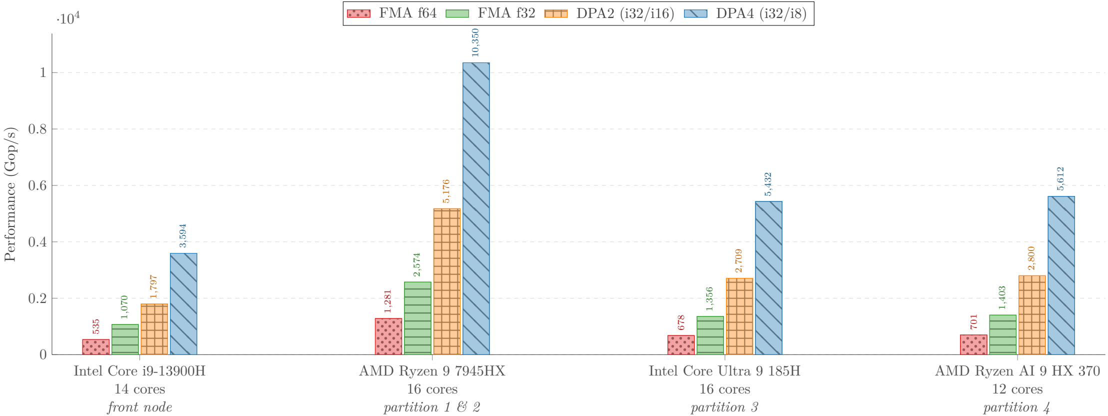
CPU Peak Performance – Multi-core

Measurements with the **cpufp** benchmark. **Higher is better**.



CPU Peak Performance – Multi-core Accumul.

Measurements with the `cpufp` benchmark. **Higher is better**.



GPU Memory Throughput

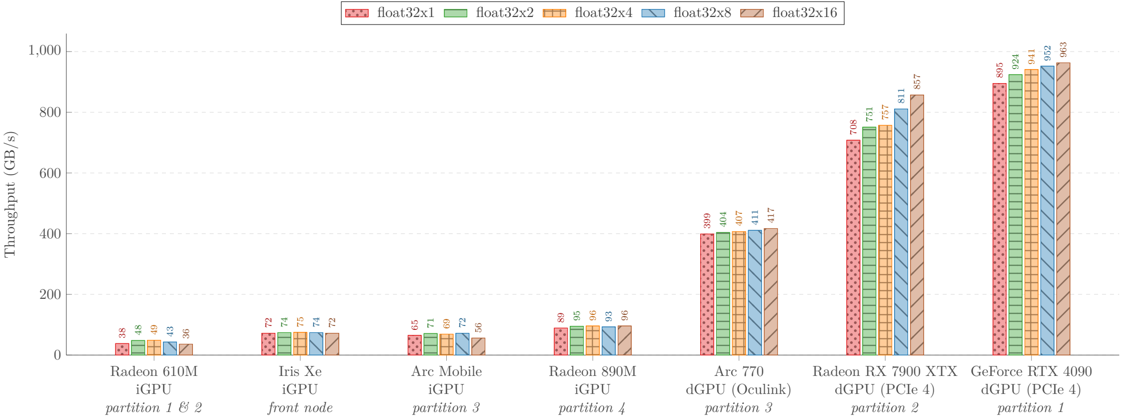
- Generally GPU can **achieve higher memory throughput than CPU**
  - Even when the memory is shared
- Throughput is measured with `clpeak`[a]
  - Commonly used to benchmark GPUs
  - Based on portable OpenCL kernels
  - Open source and available on GitHub

[a] K. Bhat et al. *clpeak*. GitHub: `https://github.com/krrishnarraj/clpeak`. 2013.

- GPUs natively support packed types
  - `float32x1`
  - `float32x2`
  - `float32x4`
  - `float32x8`
  - `float32x16`
- Help to hide instructions latency
- Can be vectorized by the GPU

GPU Memory Throughput – VRAM

Measurements with the `clpeak` benchmark. **Higher is better**.



GPU Peak Performance

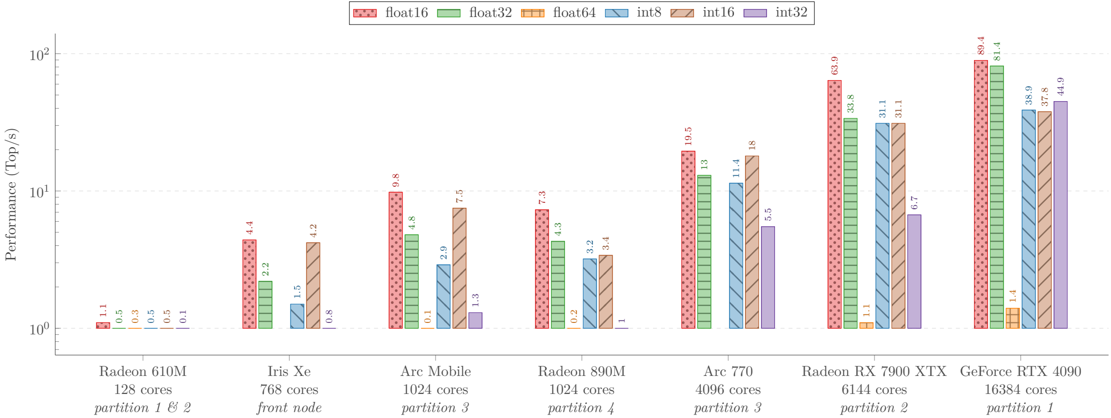
- **GPU peak is generally crazy high compared to CPUs**
  - Using Top/s (= 10^{12} op/s) instead of Gop/s
- Peak performance is also measured with `clpeak`[a]
  - Target only the general compute cores
  - Tensor and ray tracing cores are not evaluated
  - Open source and available on GitHub

[a] K. Bhat et al. *clpeak*. GitHub: `https://github.com/krrishnarraj/clpeak`. 2013.

- GPUs performance varies a lot depending on the datatypes
  - `float16`
  - `float32`
  - `float64`
  - `int8`
  - `int16`
  - `int32`
- Generally GPUs performs well on `float16` and `float32`
- Performance on `float64` and integers can be disappointing

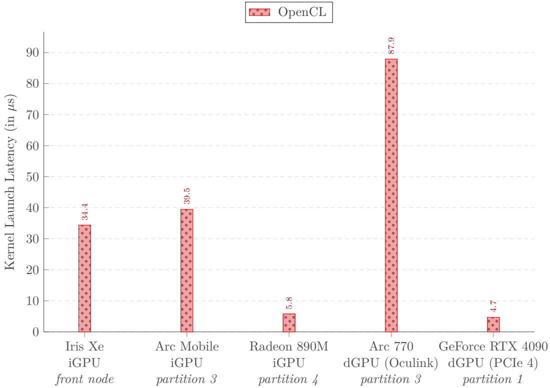
GPU Peak Performance – Shader Cores

Measurements with the `clpeak` benchmark. **Higher is better**.



GPU Kernel Launch Latency

- The kernel launch latency is the time between the GPU kernel call on CPU and the effective start of the kernel on GPU
- Why it matters?
  - Because it can be a limiting factor for applications that relies on both the CPU and the GPU
  - Typically when we want to **pipeline short kernels** between CPU and GPU



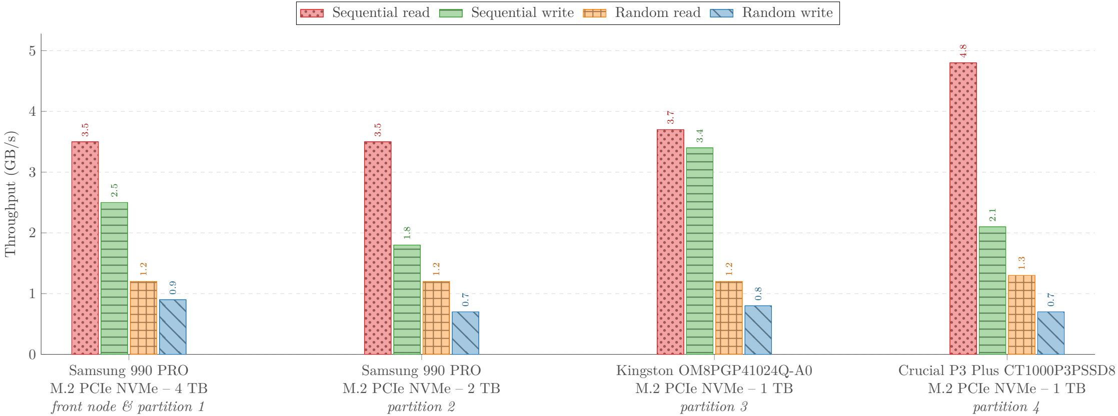
Lower is better.

SSD Performance

- Drives performance improves a lot thanks to the switch from Hard Disk Drives (HDDs) to **Solid-State Drives** (SSDs)
  - Enabling new uses: **Swapping RAM pages is much faster** than before
  - Even more true considering the **latency**
- **Sequential accesses** throughput is measured with the `dd` system command
  - The system cache is flushed
- **Random accesses** throughput is measured with the `iozone` system command

SSD Performance – Measured Throughput

Measurements with the `dd` and `iozone` commands. **Higher is better**.



References

[1] A. Cassagne, L. Lacassagne, M. Bouyer, et al. *Monolithe*. Web page: https://monolithe.proj.lip6.fr. 2022.

[2] F. Lemaitre and L. Lacassagne. *bandwidth*. GitHub: https://github.com/alsoc/bandwidth. 2017.

[3] Y. Gao et al. *cpufp*. GitHub: https://github.com/pigirons/cpufp. 2016.

[4] K. Bhat et al. *clpeak*. GitHub: https://github.com/krrishnarraj/clpeak. 2013.

[5] Adrien Cassagne, Noé Amiot, and Manuel Bouyer. "Dalek: An Unconventional and Energy-Aware Heterogeneous Cluster". (preprint). Aug. 2025. DOI: 10.48550/arXiv.2508.10481.

[6] A. Cassagne, O. Hartmann, M. Léonardon, K. He, C. Leroux, R. Tajan, O. Aumage, D. Barthou, T. Tonnellier, V. Pignoly, B. Le Gal, and C. Jégo. "AFF3CT: A Fast Forward Error Correction Toolbox!" In: *Elsevier SoftwareX* 10 (Oct. 2019), p. 100345. DOI: 10.1016/j.softx.2019.100345.